Cómputo para humanistas: Breve recorrido de la tecnología computacional y sus implicaciones sociales

Dra. Martha Irene Soria Guzmán

Para abordar la esfera de la tecnología computacional,[1] y sus implicaciones sociales, considero necesario precisar algunas cuestiones técnicas, porque si bien esta tesis no pretende ahondar en nociones especializadas de cómputo, si busca ser un puente entre estos temas, las ciencias sociales y las humanidades, en cuanto vínculo de saberes, y presentar un análisis crítico necesario desde la diversidad de disciplinas.

Para ese propósito, me referiré al antecedente inmediato del cómputo: el surgimiento de la cibernética en los años cuarenta del siglo XX. **La cibernética** se centra en comprender cómo se puede automatizar una máquina para que realice una tarea de manera más eficiente, en un proceso que se asemeje al de los organismos vivos, imitando sus comportamientos de control de cambio y comunicación (Wiener 2019). El matemático y filósofo estadounidense Norbert Wiener (1894-1964) se cuestionó ¿qué hace autónoma a una máquina? Considerado el "padre" de la cibernética, él y su equipo, observaron algunos órganos y formas de vida simples con el fin de tratar de imitar este comportamiento en las máquinas. Se concluyó que una de las características fundamentales que diferencian a los seres vivos, es la capacidad de regenerarse (por ejemplo, el caso de los reptiles y la formación de extremidades una vez perdidas, la regeneración de piel luego de una cortada, etcétera) y eventualmente de "aprender" a partir de nuevos *datos* (ibíd., p. 31).

De esta propuesta se deriva el concepto de **retroalimentación**, que habría de aplicarse en el contexto militar para rectificar la trayectoria de los proyectiles en una batalla: la trayectoria de estos se calculaba de manera mecánica para que impactara en el blanco en unas coordenadas específicas. No obstante, en el caso de un fallo, se recalculaba la trayectoria con los nuevos datos generados a partir del fallo. Esto es, una entrada *input* que considerase el resultado anterior *output*, para que el siguiente disparo —el siguiente **ciclo**— computase el fallo anterior, se modificase (figura 1) y alcanzase el objetivo (Rajsbaum y Morales 2016).

[1] En la industria, es habitual el nombre de TI (Tecnologías de la Información) para referirse específicamente a este campo, sin embargo, para términos de esta investigación se considera necesario el uso del término Tecnologías Computacionales para no referir necesariamente a la industria del cómputo y el campo empresarial y vincularlo también con el ámbito social.

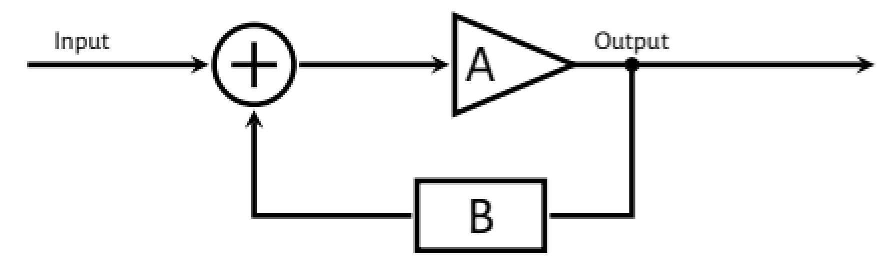


Figura 1: Ilustración para explicar el concepto de retroalimentación, donde *input* son los datos que entran y *output* un resultado que sale luego de ser computado. Los datos generados en este *output* son la *B*, que se aplicarán en el siguiente ciclo, sumando a los datos *A*, generando un nuevo

La apropiación social de los principios de la cibernética queda clara con este ejemplo, en un siglo de conflictos bélicos y conflagraciones mundiales; sin embargo, estos conceptos no son propios del siglo XX, ni Wiener el primero en pensarlos. Ya en 1843 las notas sobre la máquina analítica, de **Ada Lovelace**, dan cuenta de una idea de recursividad, o una ***reversibilidad generativa*** propuesta en tarjetas perforadas para que fuera posible repetir varias operaciones y hacer uso de bucles para operaciones más complejas, teniendo a su vez, como antecedente, el telar automático de 1801 (Zafra 2016); (Kelty 2019, p. 21).

Desde una perspectiva sociológica, la cibernética refleja las ideas funcionalistas características del siglo XX. Se trata del resultado del positivismo del siglo XIX de buscar lo determinable y reproducible, y retoma la necesidad de *control*, el deseo de producir y estudiar autómatas; hacer que las máquinas puedan parecerse a los seres vivos. Se trata, entonces, de "la ciencia ddl control de cambio" (Moreno y col. 2002, p. 31) que requiere estudiar lo que hace la máquina —**no lo que es, sino lo que hace**—.

La Bombe de Alan Turing: vueltas de engranes para contar y entregar resultados

En este contexto, el inglés **Alan Turing** (1912-1954), alumno de Alonzo Church en Princeton[2], tuvo una destacada actividad académica, pero sobre todo, un interés particular por la creación de máquinas matemáticas. Para facilitar la comprensión social de la computación que se pretende en este apartado, es necesario señalar el trabajo de Turing como el que materializaría la posibilidad de que una máquina descifrara lo cifrado por otra, en uno de los procesos automáticos más famosos de su tiempo, una vez más, en un contexto de guerra.

[2] Donde conoció al matemático húngaro-norteamericano John Von Neuman (quien a su vez conoció y trabajó con Norbert Wiener).

La Máquina Enigma era una invención alemana para cifrar letras y números en mensajes telegráficos, de comunicaciones comerciales o militares. El procedimiento para regular el cambio de estos caracteres, se llevaba a cabo mediante una máquina que utilizaba unos complicados juegos de engranajes; los principios de las calculadoras mecánicas fueron inventados por matemáticos. Según Hodges (2014, pp.202-226), durante la Segunda Guerra Mundial, matemáticos polacos lograron descifrar cómo funcionaba ese procedimiento a través del uso de la teoría de conjuntos y otros fundamentos matemáticos comunes. No obstante, el procedimiento de cifrado de la Enigma fue perfeccionado mediante el uso de motores electromecánicos y el cambio diario de la configuración de la máquina, lo cual hizo que fuera imposible para el equipo polaco calcular tantas variantes. Turing fue contratado por el gobierno de su país para descifrar el código de la Enigma y para ello, diseño una máquina electromecánica extraordinariamente compleja para su tiempo, que logró descifrar día con día información militar que contribuyó a la estrategia militar de la Segunda Guerra Mundial, empleando principios de entrada, salida, y por supuesto, **recursividad**. Las máquinas que construye el equipo de Turing para este fin, son un ejemplo de cómputo mecánico y materialización del propósito cibernético de su tiempo: una manera en la que *la materia* realiza operaciones matemáticas donde se procesa información y se obtienen resultados.

Parece una tarea difícil explicar este último concepto de forma extensa, el logro de Turing, en general, de la era del cómputo mecánico y la teoría de la computación, en el contexto de las humanidades. Sin embargo, a través de la docencia he descubierto que exponer los conceptos más fundamentales, despierta la imaginación y contribuye a una mejor comprensión de estos temas.

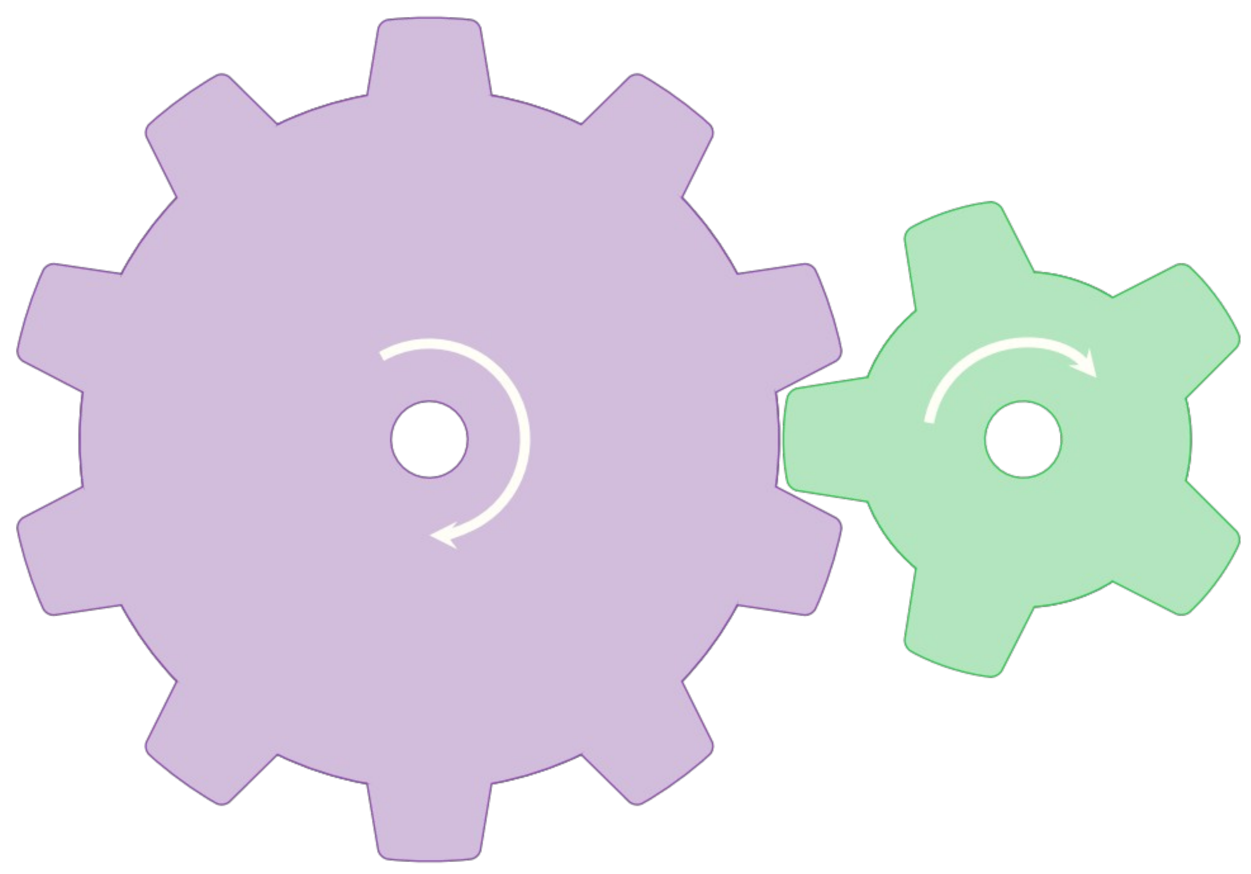


Figura 2: Ilustración que uso en mis clases para explicar el supuesto de que un par de engranes, uno con 5 dientes y otro con 10, tienen comportamientos distintos, pero vinculados en un ciclo. Mientras que el engrane más grande dará una vuelta, el engrane más pequeño, dará dos. Una forma de ejemplificar la posibilidad de materializar los ciclos y

Un ejemplo que uso en clases es cómo, de manera *mecánica*, la *materia puede aprender a contar*, lo cual es el concepto más básico de un engranaje. Imaginemos dos engranes (figura 2), uno (piñón) con 5 dientes y otro (corona) con 10 que se encuentran en movimiento conjunto. El segundo engrane gira una vez, mientras que el primero lo hace dos veces; por lo tanto, en la cuenta del número de vueltas de uno con respecto al otro, hay una diferencia de una vuelta. En un ciclo, es decir, en una vuelta, podemos hacer una división (10÷5=2) o bien una multiplicación (2×1=2).

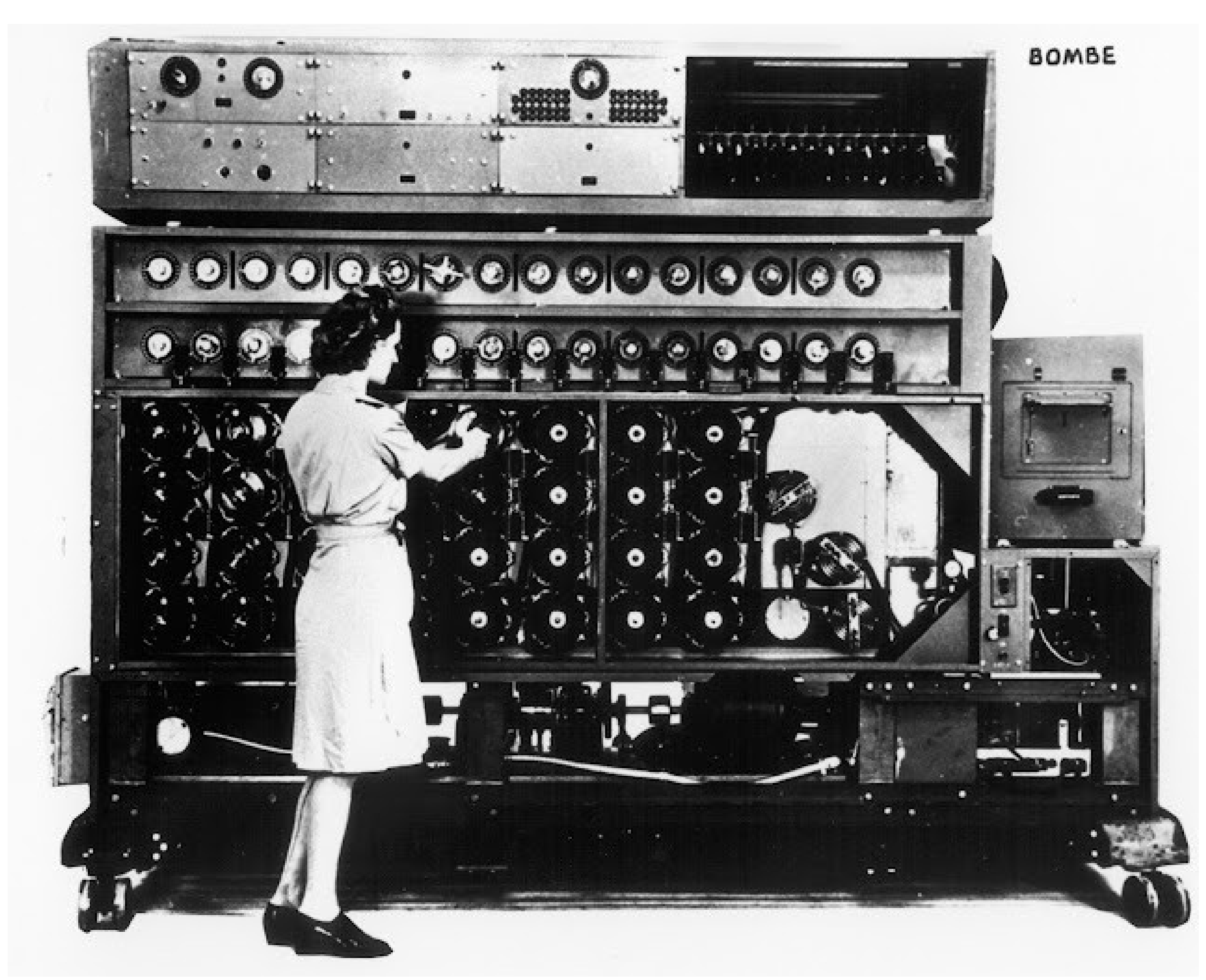


Figura 3: Una mujer aceptada para el Servicio de Emergencia Voluntario (WAVES) de la Marina de los Estados Unidos maneja la máquina de descifrado N-530 *Bombe* de la National Cash Register Company (NCR). Está ajustando la cuarta rueda conmutadora. Fuente: http://ww2images.blogspot.com/2012/12/ncr-n-530-bombe-enigma-decryption.html.

Ahora bien, para darnos una mejor idea del funcionamiento de la máquina de Turing, imaginemos unos cuantos engranes. El de una cadena de engranes supone una serie de operaciones matemáticas simples. Si cada una de las letras del alfabeto (de la *A* a la *Z*) está asociada a un número, y ese número puede representarse por la cantidad de vueltas que dan los engranes, en el último engrane de la cadena habrá un resultado. La construcción de esta máquina, implicó que Alan Turing no sólo encontrara el complejo *patrón* en los mensajes cifrados, usando los que ya habían sido descifrados por el grupo de personas encargadas de ello en *Bletchley Park*[3] y antes, el equipo polaco; significó también delimitar los movimientos, ciclos y cálculos que haría su máquina de engranes, en un sistema de transmisión de resultados

[3] *Bletchley Park* es una base militar inglesa donde se realizaron los descifrados de código de la famosa máquina alemana Enigma. Muchas de las personas que trabajaban aquí, eran mujeres. También fue cuna de las primeras computadoras digitales *Colossus*.

de una cadena de engranes a otra. Es decir, cadenas de engranes que se transmitieron el resultado de cálculos de forma recursiva. La complejidad de que las cadenas de engranajes tuvieran un resultado al final de un ciclo, y este a su vez iniciara nuevas cadenas de resultados en otra serie de engranes, —es decir, de manera recursiva— fue la innovación histórica de la llamada *Bombe* del equipo de Turing (figura 3).

En una sola década, los elementos necesarios para que *las máquinas que hacían cálculos con la vuelta de engranes* se convirtieran en la siguiente generación de máquinas tecnológicas aparecieron, utilizando los mismos principios y dando lugar al cómputo, tal y como lo conocemos en la actualidad.

A modo de continuación la idea de *hacer contar a la materia*, imaginemos el **ciclo** como el giro de los engranes que he tomado como ejemplo, pero ahora, visto en términos de un **ciclo en la electricidad**: el número de vueltas que da un electrón en un circuito para que se encienda una lámpara, —que por cierto, se parece al flujo de agua en una manguera—; para controlar este flujo, suele haber un interruptor que deja pasar o no a los electrones (figura 4).

En la era de la electrónica de los años sesenta, y debido a los **transistores**, fue posible que este flujo de electrones se regulara, interrumpiera o no, conectando un tercer flujo. Esto es, encadenando vueltas resultado de otra cadena, no de engranes como los de la *Bombe*, sino de flujos eléctricos.

La invención del transistor facilitó que la materia no solo pudiera procesar datos, sino que pudiera transmitirlos, de manera que funcionaran como compuertas físicas que permiten o no el paso, —la llave del grifo— que deja o no correr el flujo del agua. La era de los transistores resulta fundamental para el surgimiento del cómputo como lo conocemos hoy, debido a la posibilidad de controlar el flujo de energía y a las compuertas que lo permiten.

La búsqueda de materiales para fines capitalistas, afines a la modernidad y a un supuesto "progreso", llevan a la industria al descubrimiento de materiales **semiconductores**, que no conducen tan libremente la electricidad —como lo hacen los metales, conductores por antonomasia—, pero que tampoco impiden su paso —como lo hacen los materiales dieléctricos como los plásticos—. El cuarzo, por ejemplo, es un material semiconductor que se encuentra prácticamente integrado a todos los aparatos electrónicos que hay en nuestros hogares. Permite que los electrones fluyan en **ciclos** que se completan en el mismo tiempo. De esta forma, por ejemplo, pueden hacer funcionar el marcaje de un segundo en un reloj digital. La era de los **semiconductores** de los años setenta dio lugar a los modernos equipos de cómputo, lo que posibilitó las *compuertas lógicas* que permitieron la elevada integración de transistores. Se sustituyó el control y encadenamiento de los flujos eléctricos con las propiedades del silicio, y se inició una carrera

irracional y capitalista para construir cadenas diminutas de compuertas y vueltas eléctricas que cupieran en computadoras tan pequeñas, que se pudieran llevar en nuestros bolsillos,[4] con muchos de los principios maquínicos de los años treinta.

En este sentido, si unimos la posibilidad del **flujo de electrones en ciclos** y **la teoría matemática de la información** (Castro y Filippi 2010) de principios del siglo XX, cuyo concepto fundamental es que la cantidad de información contenida en un mensaje es un valor matemático bien definido y medible, podríamos preguntarnos: ¿es posible comunicarnos a partir de estos ciclos? ¿Podríamos emitir información solo con encender y apagar una luz en cada ciclo? ¿Cuál es el mínimo de información que podemos proporcionar? ¿Cuál es el mínimo de conocimiento que podemos tener de la materia?

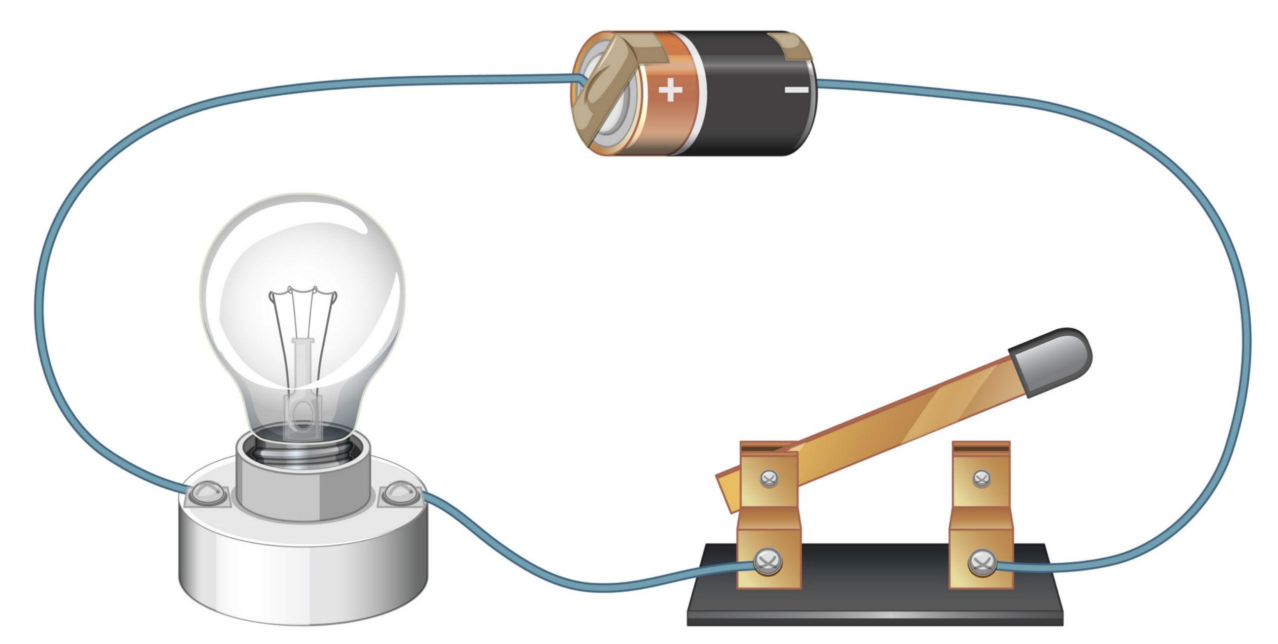


Figura 4: Diagrama que muestra un circuito eléctrico con batería, lámpara y un interruptor. Vector creado por brgfx `www.freepik.es`. `https://www.freepik.es/vector-gratis/diagrama-que-muestra-circuito-electrico-`

El *bit* y la digitalización

La digitalización, y por lo tanto, el cómputo como lo conocemos hoy en día, es posible gracias a la representación de la existencia, el estar o no estar, lo verdadero o falso, con la mínima cantidad de información, es decir, el **bit** (*binary digit*), que es la unidad mínima de información que puede ser transmitida o almacenada. En concordancia con la idea de la lámpara, si esta está encendida en un ciclo, entonces el bit tendrá el valor de 1, o *verdadero*; si está apagada, la *no existencia* será representada con un 0, un *falso*.

[4] Me refiero a los celulares o *smartphones*.

No obstante, ¿cómo es posible que, hoy en día, con solo el encendido y apagado de una *lámpara*, en un ciclo, se pueda representar tanta información? ¿Cómo es que esa información se convierte en datos o en código?

Una matriz de **bits** nos puede permitir hacer representaciones numéricas más complejas. Tomemos, por ejemplo, una cadena de 8 bits. Vamos a formarlos en posición de derecha a izquierda. Esta serie numérica permite representar un número, por ejemplo, decimal. Asignemos un valor a cada bit así: la primera posición a la derecha es 1, la siguiente es 1×2=2, y la siguiente 2×2=4, y así sucesivamente: 4×2=8; 8×2=16; 16×2=32; 32×2=64, 64×2=128...

Pongamos dicha serie, tal y como acordamos, de derecha a izquierda, y representemos, por ejemplo, el número 36, encendiendo *la lámpara* en las casillas que sumadas me den, es decir, la casilla del 32 y la casilla del 4, las cuales en este ejemplo, he marcado en negro. Para representar el 36, solo hay una posibilidad en cada ciclo:

128	64	32	16	8	4	2	1	
								36
8 bits	7 bits	6 bits	5 bits	4 bits	3bits	2 bits	1 bit	

Si ahora, asignamos un 0 en las casillas *no encendidas* y un 1 en las casillas encendidas —representadas, como ya se dijo, en color negro— entonces tendremos la representación del número 36 en binario, en un sistema de 8 bits, es decir:

128	64	32	16	8	4	2	1	
0	0	1	0	0	1	0	0	36

00100100=36

si, cada casilla representa la unidad mínima de conocimiento humano, es decir, un ***bit***, el cual puede tomar dos valores: 0 y 1, está o no está, existe o no existe. 8 bits, unidos, forman 1 *byte*, y esto, debido al avance tecnológico que permitió que las computadoras se popularizaran comercialmente, se convirtió en la referencia digital común para cuantificar la información:

8 bit = 1 B (byte)

1024 B = 1 kB (kilobyte)

1024 kB = 1 MB (Megabyte)

1024 MB = 1 GB (Gigabyte)

1024 GB = 1 TB (Terabyte)

1024 TB = 1 PB (Petabyte)

1024 PB = 1 EB (Exabyte)

1024 EB = 1 ZB (Zettabyte)

1024 ZB = 1 YB (Yottabyte)

Código e instrucciones

Ahora bien, a partir de estos conceptos, podemos pensar que este número que el número 36, representado por ceros y unos (00100100=36), está asociado a la ejecución de una acción. Ese número 36 es un **código asignado arbitrariamente por el fabricante**. Aún no nos referimos a algoritmos, programas o *software*, pero ya es posible pensar en la función del diseñador de la máquina, es decir, el fabricante que **asigna códigos** a acciones que *realizará la materia*, y que a su vez, conforma las tareas de una máquina, de una computadora; de manera que está implícito no solo un control arbitrario de un código, sino posiblemente también un proceso social de control debido al uso masivo de estos aparatos.

Esta analogía, en términos de las ciencias sociales, nos permite problematizar en cuanto a la escala de la relación entre máquinas y control en la asignación de códigos, con cadenas de letras y números que se representan posteriormente en palabras. Estas palabras pueden albergar mensajes, o documentos de todo tipo. Una cadera de caracteres también puede ser un **comando**. Por ejemplo, el famoso comando `PRINT`, que es un conjunto de instrucciones para que se *pinte* algo en la pantalla.

Por tanto, los conjuntos de comandos asignados a códigos, conforman **lenguajes de programación**, los cuales tienen diversos niveles: el más bajo nivel es muy complejo (programación en ensamblador) y un alto nivel es un lenguaje de programación muy cercano al lenguaje verbal, (como Phyton, Ruby, Haskell, etc.). La mayoría de los lenguajes de programación son una combinación de términos en inglés y matemáticas.

Ahora bien, un **algoritmo** es un conjunto de instrucciones precisas, finitas, ordenadas y claras que se utilizan para resolver un problema en el cómputo. Este algoritmo está escrito en un lenguaje de programación. Un algoritmo puede representarse mediante tres números asignados a un código por el creador de la máquina computacional. Imaginemos cuatro instrucciones asociadas a un número:

```
36: 10 20 30 50
```

O, por varios caracteres asociados a números:

```
print “Hola mundo”

print “Fin del programa”
```

En ambos casos, podemos hacer referencia a esto como el **código fuente de programación**, que es la **expresión de un algoritmo**. Además de ello, en la evolución de los lenguajes de programación, en un lenguaje de programación de bajo nivel, es más sencillo asociar acciones computacionales a números, aunque sea menos evidente para el lenguaje humano cotidiano. En un lenguaje de programación de alto nivel, en el que las palabras y las expresiones representan tareas, una expresión puede referirse a conjuntos de comandos cada vez más complejos.

Además, la evolución de los lenguajes de programación durante las últimas tres décadas del siglo XX permitió la materialización de la **programación orientada a objetos**, en la que la abstracción y complejidad de estas instrucciones permiten incluso la definición previa de entidades con *características* llamadas objetos. El lenguaje de programación en sí mismo le permite a una máquina identificar estos objetos abstractos a través de instrucciones conocidas como **constructores**.

El cómputo comercial, tal y como lo conocemos hoy en día, sigue funcionando con los mismos principios que las calculadoras mecánicas del siglo XX, solo que esta vez, se encuentran mediatizadas por muchas abstracciones y representaciones.

Incluso, el llamado aprendizaje de máquina o *machine learning* hereda del concepto de retroalimentación de la cibernética, y deviene en el de **recursividad**, un concepto abstracto que tiene que ver con lógica y matemáticas y fundamenta gran parte de lo que compone a la inteligencia artificial y sus conceptos como *trigger* o **disparador**, **condicionales** (*if, then, else*), umbrales, etcétera, que definirán el comportamiento de una posible máquina que "piense" y que lograse pasar el Test de Turing.[5]

Por ello, desde las ciencias sociales, las humanidades, y más específicamente del feminismo, resulta indispensable dejar de pensar el código fuente de programación únicamente como líneas que son un conjunto

[5] Turing escribió el texto “Computing Machinery and Intelligence” (Turing 1950), donde se origina la propuesta del *imitation game*, hoy conocido como el Test de Turing. La importancia de esta prueba, radica en que algunas personas la consideran la base de la actual Inteligencia Artificial; el punto de llegada de esa automatización. En el juego de la imitación, una persona hace preguntas que deben responder un humano y una máquina, y esta persona tendrá como objetivo discernir si las respuestas fueron formuladas por un ente que puede o no pensar. La *humanidad* de las/los participantes se juzga por las respuestas dadas, y así, propone al pensamiento como criterio de la *identidad humana*. Esto resulta relevante, pues ha derivado en diversas reflexiones éticas sobre *lo que nos hace humanidad*: el hecho tangible de que, cuando nuestra sociedad y era tecnológica vean nacer la primer máquina *pensante* que pase el Test de Turing, se habrán tomado muchas decisiones, escrito mucho código y desatado procesos, y donde la mirada desde el feminismo y la interseccionalidad, habrán resultado indispensables.

de instrucciones que controlan la operación de una máquina computacional, es decir, no debe ser reducido a un *programa que activa un mecanismo*, sino que, como diría la ingeniera Avri Doria, citada por Graciela Baroni Selaimen: "**código son bits de intencionalidad**" (Baroni Selaimen 2013, pp. 131-132). El código refleja una serie de conexiones, prácticas, procesos semióticos y cognitivos, ejercicios de poder, formas, decisiones e interacciones en el contexto social, que incluso refleja zonas de disputa geopolíticas y tecnológicas.

En este sentido, debemos considerar que se toman una serie de decisiones en prácticamente todos los procesos que involucran la tecnología computacional, incluso en el tiempo; desde cómo se piensan, cómo se conciben, la fabricación, los materiales usados, la asignación de códigos por la empresa fabricante, la escritura de algoritmos, la definición de constructores y un largo etcétera. Estas decisiones implican un componente político del que rara vez se es consciente, ya que, como diría Graciela Baroni Selaimen, "los artefactos tecnológicos son artefactos políticos, incorporan visiones de mundo y formas específicas de ejercicio del poder en varios niveles, y este ejercicio se da de manera más invisible a nivel de los códigos y de los protocolos" (ibíd., p. 131).

Y lo que es aún más complejo, con el intercambio masivo de información y datos en Internet, y la captación de los mismos por parte de empresas tecnológicas, el código no solo regula muchos de los mecanismos de comunicación de hoy, también predice comportamientos de las personas usuarias e incluso es capaz de moldearlos Amoore 2020). Eventualmente, esta información se convierte en los insumos del aprendizaje automático de máquina que promete la construcción de una inteligencia artificial. Las preguntas se extienden a ¿Qué características se tendrán en cuenta para activar los disparadores (*triggers*)? ¿Cómo y quién determinará el umbral? ¿Cómo podrán funcionar los condiciones? ¿Qué patrones están sirviendo de base a esta inteligencia artificial?

Si a todo ello, le sumamos que existe una producción de efectos que bien podrían ser elementos centrales de un nuevo *biopoder*[6] (Foucault 2011) y una extracción de datos que orientan la vida misma hacia el capitalismo y a una nueva forma de colonialismo (Couldry y Mejias 2018), entonces podríamos retomar también a Lawrence Lessig, quien dice que el código es ley, y como toda ley, trae consigo una serie de decisiones políticas que debemos observar, cuestionar y auditar. Es por ello que propongo asumir el carácter político del código y afirmar que **el código es político**.

[6] Para Foucault,"el establecimiento, durante la edad clásica, de esa gran tecnología de doble faz — anatómica y biológica, individualizante y especificante, vuelta hacia las relaciones del cuerpo y atenta a los procesos de la vida— caracteriza un poder cuya más alta función no es ya matar sino invadir la vida enteramente" (Foucault 2011, p. 169).

La decisión política de los grandes corporativos que actualmente controlan y dominan la industria tecnológica: Google, Amazon, Facebook, Apple y Microsoft (GAFAM) y que proporcionan a una gran cantidad de personas en el mundo todos los servicios que regulan nuestras vidas, ha sido, y seguirá siendo, **el cierre de su código fuente**.[7] Estas empresas que hoy controlan y conocen gran parte de las actividades de la población mundial tomaron la decisión de no permitir que nadie más que un grupo minúsculo de personas conozcan el código fuente de programación. Su decisión ha sido ocultar, cerrar y ofuscar en aras de obtener un beneficio económico.

A continuación, expondré mi opinión sobre el cierre del código fuente de programación y sus implicaciones en cuanto *software*.

El código es político: el cierre de un saber-hacer

Se ha mencionado que la cibernética es el origen de la computación moderna, la cual requiere que las máquinas puedan seguir una serie de instrucciones, ordenadas y finitas, para llevar a cabo una acción, lo que actualmente se conoce como algoritmos, y que luego fueron representados en el **código fuente de un *software***. Gracias al trabajo de la comunidad científica, específicamente de personas versadas en matemáticas y física, se crearon los primeros programas de computación[8] a finales de la década de 1950, lo cual representó un importante cambio de paradigma en el desarrollo y ejecución de la técnica. El surgimiento del *software*, sin lugar a dudas, **estableció la dirección** de la tecnología computacional que se utiliza en la actualidad.

Ahora bien, para comprender mejor qué es el *software* o programa de computadora, y cómo se relaciona con la técnica, retomaré la metáfora que usa Stallman (2004), al compararlo con una receta de cocina. El *software* es un conjunto de instrucciones meticulosamente detalladas para la solución de un problema específico, que puede ir desde hacer una suma hasta escribir una carta, crear un dibujo vectorial o editar un video. Las recetas antes mencionadas están redactadas con una sintaxis que proviene, tanto del inglés como de las expresiones matemáticas; podríamos denominarlo lenguajes formales,[9] los cuales son empleados para la programación.

[7] Vale la pena apuntar y problematizar este punto, ya que estas industrias son de las que más código fuente publican y conviven con ese modelo de código abierto, sin embargo, contribuyen y realizan proyectos de código visible, a la par de que toman código hecho por la comunidad y la cierran para usarlo como parte de sus servicios que luego, extraen datos e información de las poblaciones. Considero que el punto aquí, es que se benefician del trabajo de una comunidad, en aras de sus ganancias económicas, y no así para beneficio social.

[8] Programas de cómputo y *software*, serán usados como sinónimos en este trabajo.

[9] Ejemplos de lenguajes formales de programación son: Phyton, C++, Java, Javascript, Ruby, Cocoa, Cobol, etcétera.

Bernard Stiegler también usa la metáfora de la receta de cocina cuando establece una conexión entre la técnica y la tecnología que integra la ciencia. Para Stiegler, la cocina está relacionada con la producción y transformación de materiales en productos, y, por lo tanto, es fácilmente reconocida como técnica, es decir, como ya se ha mencionado: saber-hacer (Stiegler 2002, p. 145).

A mediados de la década de los setenta, era muy habitual entre la comunidad de programadores del MIT en Estados Unidos, intercambiar *software* y, de esta forma, solicitar y ofrecer parte del código fuente para mejorarlo de forma conjunta. No obstante, la lógica capitalista y el refuerzo del neoliberalismo influyeron en las prácticas computacionales de la época y a finales de esa misma década, algunas empresas pioneras de computación prohibieron el acceso al código fuente, su libre compartición y establecieron un costo para la distribución de su *software* exclusivamente como programas ejecutables o aplicaciones (archivos binarios). Algunas computadoras modernas de la época comenzaron a tener su propio sistema operativo para el cual se requería firmar un acuerdo de confidencialidad para obtener una copia ejecutable (Stallman 2004, p. 21).

Gradualmente, el uso y la distribución del código fuente se transformó en su privatización, volviéndolo cerrado y desembocando posteriormente en la creación y uso de patentes de *software* en los Estados Unidos. Poco tiempo después, la venta de *software* se convirtió en una muy rentable actividad comercial para las corporaciones. Así comenzó la era de la comercialización del *software* sin código fuente abierto (lo que hubiera permitido conocerlo y estudiarlo), por el contrario, se trataba de un **código cerrado** que representó un velo y que impedía saber cómo fue hecho. Esto marcó el inicio de la era en la que se empezó a utilizar una caja cerrada y desconocida como herramienta tecnológica. Si atendemos a la definición de técnica como saber-hacer que se planteó al comienzo de este capítulo, el cierre del código del *software* supone entonces **una nueva forma de ocultación y privatización del saber-hacer, así como un ejercicio de poder más**. Si la tecnología digital del siglo XXI, que requiere el uso de *software*, tiende a *ocultar* el código fuente, esto nos alejaría necesariamente cada vez más de la técnica. En consecuencia, la falta de visibilidad y de acceso para modificar el código es una problemática relevante que merece reflexión y análisis, pues el ocultamiento de la *receta de cocina* dificulta e impide también el estudio y la práctica de la técnica y ciertos saberes.

Estas y otras consecuencias (por ejemplo, que el *software* de código cerrado es propiedad privada) han motivado a un grupo de personas[10] a llamarle ***software* privativo** para subrayar que *priva* libertades y *privatiza* el conocimiento. Lo cierto es que se ha convertido en un *software* comercial y

[10] Es importante señalar que este grupo de personas fueron usuarios del sur de América Latina, particularmente en Argentina, fue el filósofo Enrique Chaparro quien acuñó el término: *software privativo*.

hegemónico, muy utilizado, normalizado y poco cuestionado, que replica los principios del sistema económico y político del capitalismo neoliberal.

Todo lo que se ha mencionado hasta este punto me lleva a afirmar que la apropiación tecnológica computacional de las mujeres se ve dificultada, no solo por el alejamiento de ciertas técnicas especializadas vinculadas con las máquinas y la modernidad en el capitalismo, como se explicó al inicio de este capítulo, sino que también en el caso más reciente de las tecnologías digitales y de información, que ocultan los procesos que lleva a cabo el equipo de cómputo, —como programas que corren en paralelo sin que lo sepamos— y la mediación que representa el ocultamiento del código, que impide conocer cómo son hechos estos programas, qué hacen, para qué lo hacen y sobre todo, imposibilita su modificación y adaptación. Si bien, esto último afecta a todas las personas usuarias, la generización de la tecnología deWajcman (2006) de la que ya se habló en el apartado *Estudios de mujeres y tecnología computacional [fuera de este extracto]*, sitúa a ciertos grupos de mujeres en una esfera de alejamiento aún mayor. Aunque el código puede ser modificado y adaptado, hay muchos otros problemas que surgirán incluso en un proyecto de código abierto, los cuales no serán tratados a fondo en esta tesis. Estos problemas pueden variar, desde no comentar el código que se realiza, impidiendo que otras personas conozcan los procedimientos, hasta una complejidad importante debido a la cantidad de datos que se manejan. Aunque la apertura, en términos recientes, no es una condición necesaria para la apropiación tecnológica, yo considero que sí es necesaria.

Apuntes para problematizar la privatización del *software*

En esta hipermediación digital, que cada vez afecta más áreas de nuestra existencia, están en particular en juego nuestra vida, nuestra intimidad y nuestros deseos, ya que si bien, ofrece inmediatez, comodidad, conexión con personas queridas y circulación de afectos, todo esto es a cambio de datos precisos y diferenciados por grupos de población. Si extendemos la reflexión al campo de las ciencias sociales, también está en juego la organización y la lucha social, como lo ha sido la nueva ola feminista, por ejemplo, cuya esfera de acción se encuentra, en gran medida, en las redes sociodigitales, donde además se subraya la polarización de posturas ideológicas.

En este contexto, el *ciberpunk* de la ciencia ficción planteó acciones de resistencia tecnológica dentro de una posible distopía, con el *do it yourself* y con la figura hacker que apelaban a la necesidad de comprender cómo funcionaba todo para luego *romperlo* y crearlo de nuevo. Años después, estos imaginarios se vieron reflejados a través de ciertas prácticas y políticas *hacktivistas*, que podrían darnos algunas claves para deconstruir la distopía.

En esta *distópica realidad* presente, la postura hacker se pregunta: ¿de quién es el conocimiento?, ¿este debe ser reconocido como valioso, solo cuando viene de sujetos autorizados por su función en la estructura capitalista? En la búsqueda de una sociedad justa, para el *hacktivismo* (hacker + activismo), la lucha por el conocimiento en nuestra época se asemeja a la de la tierra, o a la del capital, es decir, una lucha entre propietarios, que defienden o amplían su reivindicación de la propiedad privada; y entre los desposeídos, que luchan por ampliar o defender la propiedad pública. En consecuencia, si los agricultores luchan contra la expropiación de la tierra, los obreros, contra la expropiación de su trabajo, las personas hackers lucharán por socializar los flujos de información y del conocimiento (Wark 2004) a través del hacktivismo. Algunos *hacktivistas* ven en las empresas tecnológicas una amenaza a las herramientas de producción inmediatas, en tanto que el poder del mercado se apropia de la información y del aprendizaje libre, fruto de una red de conocimiento. De este modo, se produce una especie de escasez artificial, debido a la acumulación de información. Este poder del mercado, influye incluso en las definiciones que el Estado hace sobre la representación de la propiedad en lo digital (ibíd., p. 96), desencadenando en el endurecimiento de leyes restrictivas de propiedad intelectual, que perjudican al bien común.

La idea del *ciberespacio* que ofrece el ciberpunk, también previó la amplificación de los poderes y las desigualdades del mundo físico, donde ciertos grupos sociales se verían más afectados que otros. La cada vez mayor necesidad de estar hiperconectados en nuestra vida en *cautiverio* nos ha permitido (re)conocer a Internet como este espacio que, para ser habitado, requiere el pago de una renta tecnológica, como señaló el filósofo latinoamericano Bolívar Echeverría: “si llamamos renta de la tierra al dinero que el terrateniente recibe por el uso de su tierra, podemos llamar también renta tecnológica al dinero que el propietario tecnológico recibe por el uso de ‘su’ tecnología” (Echeverría 2005, p. 19) en este caso, a través de los datos que otorgamos a cambio de una cuenta *gratuita* en alguna de las cinco grandes empresas que ostentan el monopolio tecnológico GAFAM. La búsqueda de nuevas estrategias para construir sistemas más éticos e inclusivos, donde el acceso y uso libre del conocimiento sea horizontal, así como imaginar y soñar *otros modos* de tecnologías posibles, resulta vital si pensamos en Internet como un territorio en disputa (Lechón Gómez y Ramos Muñoz 2019).

Para encontrar una fisura por donde se cuele la luz en esta distopía y dimensionar lo grave y complejo de lo que ocurre en la esfera digital, propongo mirar al cierre de código de programación, como similar al *secreto industrial* que responde a la lógica capitalista de lo que algunos llaman *la sangre del sector privado*, es decir, la propiedad intelectual. No debemos olvidar, además, que lo que está escrito en el código fuente de programación se convierte en "ley" (Lessig 2009) del *software*, los dispositivos que lo ejecutan, y por supuesto del ciberespacio en el que habitamos, de tal

manera que sin la posibilidad de saber exactamente qué hace la parte *lógica* e nuestros dispositivos, se oculta también una técnica, un *saber-hacer* y, por lo tanto, una forma de conocimiento. La postura política de Stallman en contra del cierre y privatización del conocimiento, ha permitido acuñar términos como el de *copyleft*, a manera de juego de palabras entre 'derecho a la copia' e 'izquierda', como un *hack* o truco que trastoque el sistema maximalista de *todos los derechos reservados* de la propiedad intelectual. No debemos olvidar que la aparente utopía de un conocimiento libre de la escasez y la producción libre de conocimiento, sucedió en este *mundo hacker* al inicio de la era del cómputo, donde los primeros programas de computación se compartían de manera libre y se mejoraban entre varias personas.

Una de las más valiosas propuestas que el *hacktivismo*, derivado de la cultura hacker, nos ofrece es la importancia de cuestionar cómo funcionan las cosas y encontrar en ello las posibilidades de hacer frente al sistema, lo cual nos puede llevar a cuestionarnos: ¿quién produce y cómo funciona exactamente esa tecnología que entra en nuestras vidas, en nuestros cuerpos y que conoce nuestras emociones?

Algunas propuestas se centran en poner especial atención en la educación digital, la alfabetización mediática e informacional, y en general, a la enseñanza de saberes digitales, con el objetivo de reducir la dependencia generalizada de los equipos de cómputo, y el desconocimiento del funcionamiento de las tecnologías digitales.

Aunque la educación es un elemento fundamental para el funcionamiento de una sociedad más justa e igualitaria, también hemos podido comprobar cómo la educación tradicional ha legitimado ciertas desigualdades y ha sido usada como justificación para la discriminación de grupos desfavorecidos, argumentando que estos *crecen de educación*. En las comunidades pobres, solo el 2 % de los estudiantes logran terminar sus estudios universitarios. A menudo, las familias envían a los hijos varones a la escuela y no a las hijas. Las clases altas tienen una mayor probabilidad de acceder a la educación durante un mayor número de años. Si tenemos en cuenta que el rendimiento escolar depende en gran medida del apoyo familiar y que estas condiciones suelen ser escasas en los grupos periféricos, podemos ver cómo el acceso a la educación no es suficiente para lograr una vida con mayores oportunidades para las personas en situación de precariedad.

Si siguiéramos el razonamiento de una posible propuesta hacker, tendríamos que cuestionar la educación normativa y tradicional, así como algunos modelos universitarios donde se forman seres alienados del sistema capitalista, y se inhibe la postura crítica, favoreciendo una formación tecnocrática.

Una educación hacker ponderaría las prácticas de la clase trabajadora para reivindicar el ingenio y las habilidades desarrolladas para habitar el mundo,

es decir, el conocimiento práctico que es útil para la vida (Wark 2004, p. 37), lo cual incluye el saber de las mujeres que por siglos ha sido considerado subalterno. Por esta razón, pensar en una posible apropiación tecnológica, va más allá de *educar*, pues requeriría trascender la jerarquía de la persona que *educa* y otra que *recibe* el conocimiento, esto es: romper el binomio de la persona que sabe más y que *ilumina* a quien no sabe. La supuesta solución basada en *educar a las masas*, culpa directa y exclusivamente a las personas usuarias de *no saber*, de *no leer*, y de *aceptar términos y condiciones* sin cuestionamientos, haciéndoles responsables de una decisión para la que la mayoría de las veces no tienen alternativa, porque ¿qué opciones tecnológicas inmediatas tiene una mujer de alguna comunidad indígena que requiere de comunicarse con sus seres queridos a la distancia, si no es la de recurrir a aplicaciones —mañosamente— gratuitas incluidas en el plan de datos de su compañía telefónica?

Mención especial merece la apropiación tecnológica a través de los movimientos *maker*, los clubs de reparación, y los esfuerzos individuales por prolongar la vida de los aparatos y luchar contra la obsolescencia programada, la cual se refiere a que los aparatos tecnológicos están fabricados de manera intencional para que su vida útil sea de pocos años, aumentando el consumo, la basura tecnológica y la contaminación ambiental.

Particularmente la cultura de la reparación, el armado de equipos de cómputo, la cultura del reuso, así como el uso de *hardware* libre, en combinación con el *software* libre en su parte lógica, podría ser un camino seguro y viable, aunque no sencillo, ni cómodo, ni fácil, para arar el camino de la libertad tecnológica, con miras a una posible emancipación y salida para nuestra imaginaria distopía. Pero como parte del sendero que tendremos que recorrer juntos, habrá que cuestionarnos un tema complejo: la idea de *facilidad* y de *inmediatez*. De ahí la importancia de problematizar los modelos cognitivos que suponen las tecnologías en el mercado estrechamente ligados a la lógica capitalista, que nos oculta los procesos que ocurren detrás, que los vuelve mecanismos desconocidos y lejanos al entendimiento de las y los usuarios finales, pero que sin duda son cómodos, fáciles y completamente solucionistas.

En torno a los debates actuales de la propiedad intelectual y su operatividad en el derecho de autor, —que se retomaron incluso en la pandemia, con la dificultad de acceder legalmente a archivos, materiales y herramientas en la esfera digital— por más esfuerzos que se hagan para avanzar hacia la debilitación del excesivo poder de mercado de las empresas, no habrá cambio significativo mientras no se discuta, o incluso, se llegue a la abolición del régimen de propiedad intelectual mundial (Rikap 2021). Mientras sigamos creyendo que la regulación del ciberespacio es la solución, o que solo es posible la dicotomía de control, ya sea del Estado, o de las empresas, y no abordemos lo inaudito y contra derechos humanos que significa el

código cerrado, o cuestionemos lo aparentemente incuestionable: el quiebre del secreto industrial; seguiremos pensando que los términos y condiciones que aceptamos son el último fin, cuando en realidad, solo son la punta del iceberg de una problemática profunda que es un complejo mecanismo de poder y control, que no se reduce a la esfera digital, sino que también incluye ámbitos sociales, políticos y económicos, y nos sumerge en una especie de *Edad Media tecnológica*.

Bibliografía general

Artículos

Castro, Carlos y Luis Filippi (9 de nov. de 2010). “Modelos Matemáticos de Información y Comunicación, Cibernética (Wiener, Shannon y Weaver): Mejorar La Comunicación es el Desafío de Nuestro Destino Cultural”. En: *Re-Presentaciones: Periodismo, Comunicación y Sociedad 6*, págs. 145-161. ISSN: 0718-4263.

Couldry, Nick y Ulises A. Mejias (2 de sep. de 2018). “Data colonialism: Rethinking big data’s relation to the contemporary subject”. En: *Television & New Media 20.4*, págs. 336-349. ISSN: 1527-4764. doi: https://doi.org/10.1177%2F1527476418796632. url: https://journals.sagepub.com/doi/abs/10.1177/1527476418796632.

Echeverría, Bolivar (30 de sep. de 2005). “‘Renta Tecnológica’ y Capitalismo Histórico. Conferencia en el Fernand Braudel Center de la Universidad de Binghamton. 4-Dic-1998”. En: *Mundo Siglo XXI. Revista del Centro de Investigaciones Económicas, Administrativas y Sociales del Instituto Politécnico Nacional 2*, págs. 17-20. ISSN: 1870-2872. url: https://repositorio.flacsoandes.edu.ec/handle/10469/7399.

Lechón Gómez, Domingo Manuel y Dora Elia Ramos Muñoz (20 de dic. de 2019). “¿Es Internet un territorio? Una aproximación a partir de la investigación del hacktivismo en México”. En: *Economía Sociedad y Territorio*, págs. 273-301. ISSN: 2448-6183. doi: 10.22136/est20201507. url: https://www.scielo.org.mx/scielo.php?pid=S1405-84212020000100903&script=sci_abstract&tlng=pt (visitado 30-06-2022).

Rajsbaum, Sergio y Eduardo Morales (2016). “Norbert Wiener y el origen de la cibernética”. En: *Revista de la Academia Mexicana de Ciencias 67.1*, págs. 6-11. url: https://www.amc.mx/revistaciencia/images/revista/67_1/PDF/Presentacion.pdf.

Rikap, Cecilia (24 de ene. de 2021). “El poder político en la era digital”. En: Ámbito. Url: https://www.ambito.com/opiniones/google/el-poder-politico-la-era-digital-n5164806 (visitado 30-06-2022).

Turing, Alan Mathison (27 de oct. de 1950). “Computing Machinery and Intelligence”. En: *Mind* 59.236, págs. 433-460. ISSN: 0026-4423. url: http://www.jstor.org/stable/2251299 (visitado 08-01-2022).

Capítulos de libros

Baroni Selaimen, Graciela (2013). “Mujeres desarrolladoras de tecnologías —el desafío de las historias invisibles que viven entre ceros y unos”. En: *Internet en código femenino. Teorías y prácticas*. Ed. por Graciela Natansohn. Futuribles. La Crujía Ediciones. Cap. 3, págs. 123-136. ISBN: 978-987-601-205-8. url: https://lacrujia.com.ar/#/detalles/99 (vid. pág. 6).

Libros

Amoore, Louise (1 de mayo de 2020). *Cloud ethics: Algorithms and the attributes of ourselves and others*. Duke University Press. 232 págs. ISBN: 978-1-4780-0831-6. url: https://www.dukeupress.edu/cloud-ethics (vid. Pág. 6).

Hodges, Andrew (10 de nov. de 2014). *Alan Turing: The Enigma: The book That Inspired the Film* The Imitation Game. Updated Edition. Con pról. de Douglas Hofstadter. Princeton University Press. 768 págs. ISBN: 978-0-691-16472-4. url: https://press.princeton.edu/books/paperback/9780691164724/alan-turing-the-enigma.

Kelty, Christopher M. (2019). *Two bits: Trascendencia cultural del software libre*. Ed. por Florencio Cabello. 458 págs. ISBN: 978-84-9888-881-2. url: http://bdjc.iia.unam.mx/items/show/412.

Lessig, Lawrence (2009). *El Código 2.0*. Mapas 24. Traficantes de Sueños. 560 págs. ISBN: 978-84-96453-38-8. url: https://traficantes.net/libros/el-codigo-20.

Moreno, Juan Carlos y col. (2002). *Manual de iniciación pedagógica al pensamiento complejo*. Comp. por Marco Antonio Velilla. Corporación para el desarrollo Complexus. 257 págs. url: https://www.sesge.org/images/docs/manual_iniciacion.pdf.

Stallman, Richard Matthew (17 de dic. de 2004). *Software libre para una sociedad libre*. Trad. por Jaron Rowan, Diego Sanz Paratcha y Laura Trinidad. Con intr. de Lawrence Lessig. Mapas. Traficantes de sueños. 319 págs. ISBN: 978-84-933555-1-7. url:

https://traficantes.net/libros/software-libre-para-una-sociedad-libre.

Wajcman, Judy (2006). *El tecnofeminismo*. Feminismos 88. Ediciones Cátedra. 198 págs. ISBN: 84-376-2317-0.

Wark, McKenzie (10 de abr. de 2004). *A hacker manifesto*. Harvard University Press. 208 págs. ISBN: 9780674015432. url: https://www.hup.harvard.edu/catalog.php?ISBN=9780674015432&content=reviews.

Wiener, Norbert (8 de oct. de 2019). *Cybernetics or Control and Communication in the Animal and the Machine*. Reissue Of The 1961 Second Edition. Con pról. de Doug Hill y Sanjoy Mitter. 3. a ed. MIT Press. 352 págs. ISBN: 9780262537841. url: https://mitpress.mit.edu/9780262537841/cybernetics-or-control-and-communication-in-the-animal-and-the-machine/.

Zafra, Remedios (2016). (h)adas Mujeres que crean, programan, prosumen, teclean. Voces/Ensayo 186. Páginas de Espuma. 288 págs. ISBN: 978-84-8393-555-2. url: https://paginasdeespuma.com/catalogo/hadas/.

Libros en varios tomos

Foucault, Michel (2011). *Historia de la sexualidad I: La voluntad de saber*. 3.ª ed. Vol. 1. 4 vols. 200 págs. ISBN: 978-607-03-0292-3.

Stiegler, Bernard (2002). *La técnica y el tiempo. El pecado de Epimeteo*. Trad. por Beatriz Morales Bastos. Vol. 1. 3 vols. Editorial Hiru. 412 págs. ISBN: 84-95786-26-5.

Sitios de Internet

Stallman, Richard Matthew (2002). *On Hacking*. url: https://stallman.org/articles/on-hacking.html (visitado 17-07-2018).